Remarks

Applicants respectfully request reconsideration of the present application in view of the foregoing amendments and the following remarks. Claims 1-11, 13-15, 17, 22, 23, 26-32, 34-37, 39-41, and 43 are pending in the application. No claims have been allowed. Claims 1, 15, 31, 34, 39, and 43 are independent. Claims 1, 13-15 have been amended. Claims 12 and 18-21 have been canceled without disclaimer and without prejudice to pursuing in a continuing application.

Cited Art

The Action cites Gordon et al., U.S. Patent No. 6,560,774 ("Gordon") and Burger et al., U.S. Patent Publication No. 2004/0268327 ("Burger").

Burger is Not Prior Art Under § 103(a)

U.S. Patent Application No. 10/628,054 (the present Application) and Burger (U.S. Patent App. No. 10/610,692, Pub. No. 2004/0268327) were, at the time the invention of Application 10/628,054 was made, owned by Microsoft Corporation. Therefore, under 35 U.S.C. 103(c), Burger, as a patent publication under § 102(e), cannot be used in a rejection under 35 U.S.C. 103(a) against the claims of this application. See MPEP § 706.02(l)(2).

Initialed Form 1449 Not Received

On February 12, 2009, Applicants submitted an Information Disclosure Statement listing two patent references and six non-patent references. Applicants have not yet received an initialed 1449 form for this IDS submission. Applicants respectfully request that the Examiner provide the initialed 1449 form for this IDS submission. See MPEP § 609 ("An information disclosure statement filed in accordance with the provisions of 37 CFR 1.97 and 37 CFR 1.98 will be considered by the examiner assigned to the application.").

Claim Rejections under 35 U.S.C. § 102

The Action rejects claims 1-12, 15, 17-23, 26, 27, 30-32, 34-37, 39-41, and 43 under 35 USC 102(a) as being anticipated by Gordon.

Claims 1-11 are Allowable Over Gordon

Claim 1 recites one or more computer-readable media with computer-executable instructions for implementing a software development architecture comprising, in part:

> a code generator operable to generate code targeted for a plurality of execution architectures;
>
> wherein the code generator constructs one or more software development components of a plurality of different software development tools using the software development scenario-independent intermediate representation format, the one or more exception handling models operable to support the plurality of programming language specific exception handling models for the plurality of different source languages, and the type system operable to represent the plurality of different source languages; and
>
> wherein the code generator further creates the plurality of different software development tools using the one or more software development components and the software development architecture.

Claim 1 has been amended, in part, with language similar to dependent claim 12. In addition, see the Application at page 7, line 25 to page 8, line 7 and page 37, line 8 to page 39, line 2, and Fig. 13.

Gordon does not teach or suggest the above-cited language of claim 1.

Gordon recognizes the following problem, "[a] difficulty arises in intermediate language-type models in that, in some circumstances, the execution engine needs to run untrusted code." (Gordon, 24-26.) As a proposed solution to this problem, Gordon describes "a verifier to check intermediate language code." (Gordon, 1:45-46.) Gordon states that:

> In one embodiment, a computer-implemented method first verifies metadata of an intermediate language code for consistency and accuracy, and then verifying the intermediate language code for consistency and accuracy. This latter part in one embodiment is accomplished by performing first a syntactic check of the intermediate language code, and then a semantic check of the intermediate language code.
> (Gordon, 1:46-53.)

Thus, Gordon describes a code verification method that verifies code in several different ways to ensure consistency and accuracy of untrusted code. At no point does Gordon teach or suggest constructing (or generating) software development tools, much less the above language of claim 1 reciting "components of a plurality of different software development tools [constructed] using the software development scenario-independent intermediate representation format, the one or more exception handling models operable to support the plurality of

programming language specific exception handling models for the plurality of different source languages, and the type system operable to represent the plurality of different source languages."

The Examiner argues that Gordon describes this language, citing Gordon's description of an Execution Engine at col. 25, lines 25-55. Action, page 5, with respect to claim 12. In addition, the Examiner cites to Gordon , Figure 23 and related text referring to JIT compilers. Action, page 9, with respect to claim 31. Applicants respectfully disagree.

First, at col. 25, Gordon describes the COM+ Execution Engine as follows, "Execution Engine (EE) that manages the execution of source code compiled into Intermediate Language (IL), OptIL, or native machine code. All code based on COM+ IL executes as managed code, that is code that runs under a 'contract of cooperation'. The environment in which the EE operates is referred to as the COM+ Runtime environment, or simply COM+ Runtime." Gordon, col. 25, lines 13-19. As Gordon describes at col. 25, the Execution Engine is a runtime environment for executing IL or native code. In addition, the Execution Engine can function as a JIT compiler supporting on-the-fly conversion of IL to native code. Gordon, col. 25, lines 51-61. Nowhere does this section of Gordon teach or suggest, "wherein the code generator constructs one or more software development components of a plurality of different software development tools," as recited by claim 1. Furthermore, the remaining disclosure of Gordon does not teach or suggest this language.

Second, Gordon describes Fig. 23 as follows, "The Execution Engine creates an environment for code execution called the Virtual Execution System, which is shown in FIG. 23. In most cases, source code is compiled into IL, and the IL is loaded and executed on-the-fly using one of the JIT compilers to convert the IL to native code." Gordon, col. 27, lines 12-16. Therefore, this section of Gordon describes compiling and executing IL code, including using a JIT compiler. This section of Gordon does not teach or suggest, "wherein the code generator constructs one or more software development components of a plurality of different software development tools," as recited by claim 1. Instead, Gordon describes loading intermediate language code and executing the intermediate language code using a JIT compiler to convert the intermediate language code to native code.

Furthermore, the above sections of Gordon do not teach or suggest, "wherein the code generator further creates the plurality of different software development tools using the one or more software development components and the software development architecture," as recited

by claim 1. The Execution Engine and Virtual Execution System described by Gordon compile and execute code. While these systems of Gordon use software development tools (e.g., JIT compilers), they do not create (or generate) software development tool components or software development tools.

For at least the above reasons, Gordon does not teach or suggest the above-cited language of claim 1. Therefore, claim 1 should be in condition for allowance.

Dependent claims 2-11 should be allowable for at least the reasons discussed above with regard to claim 1.

Claims 15, 17, 22, 23, 26-27, and 30 are Allowable over Gordon

Amended claim 15 reads as follows (emphasis added):

> A method of creating a target software development tool, the method comprising:l
>
> receiving at least one computer-readable specification specifying functionality specific to one or more software development scenarios, *wherein the at least one computer-readable specification specifies the following software development scenario functionality of the target software development tool:*
>
> *target processor execution architecture;*
> *type checking rule set;*
> *managed execution environment;*
> *input programming language or input binary format; and*
> *compilation type*;
>
> *creating at least one software development component for the software development tool from the at least one specification*;
>
> integrating the at least one software development component for the software development tool into a software development scenario-independent framework; and
>
> *compiling the at least one software development component and framework to create the target software development tool*;
>
> wherein the computer-readable specification comprises functionality for processing an intermediate representation format capable of representing a plurality of different programming languages; and
>
> wherein the intermediate representation format comprises one or more exception handling models capable of supporting a plurality of programming language-specific exception handling models for the plurality of different programming languages.

Claim 15 has been amended with language similar to dependent claims 18-21. In addition, see the Application at page 8, lines 10-24 and page 12, lines 5-20.

As discussed above with regard to claim 1, Gordon does not teach or suggest, "compiling the at least one software development component and framework to create the target software development tool," as recited by claim 15. Applicants note that this language is not addressed by the Examiner in the Action's rejection of claim 15. Action, page 6.

Furthermore, Gordon does not teach or suggest, "receiving at least one computer-readable specification specifying functionality specific to one or more software development scenarios, wherein the at least one computer-readable specification specifies the following software development scenario functionality of the target software development tool: target processor execution architecture; type checking rule set; managed execution environment; input programming language or input binary format; and compilation type," as recited by claim 15. Regarding this language, partially incorporated from dependent claims 18-21, the Examiner cites to Gorodon Fig 23, JIT compiler and related text, and Fig. 2, VB, VC++, Other. Action, page7, with regard to claims 18-21. Applicants respectfully disagree. These sections of Gordon describe a Virtual Execution System that uses a JIT compiler (Fig. 23), and compiling multiple source languages into an IL (Fig. 2). None of these sections of Gordon teach or suggest a specification that "specifies the following software development scenario functionality of the target software development tool: target processor execution architecture; type checking rule set; managed execution environment; input programming language or input binary format; and compilation type," as recited by claim 15.

For at least these reasons, Gordon does not teach or suggest the above-cited language of claim 15. Therefore, claim 15 should be in condition for allowance.

Dependent claims 17, 22, 23, 26, 27, and 30 should be allowable for at least the reasons discussed above with regard to claim 15.

Claims 31 and 32 are Allowable over Gordon

Claim 31 recites (emphasis added):

> *integrating software development components comprising one or more characteristics of the target software development tool into the common framework; and*
>
> *creating the target software development tool from the integrated common framework.*

For at least the reasons discussed above with regard to claim 1, Gordon does not teach or suggest the above-cited language of claim 31. Therefore, claim 31 should be in condition for allowance.

Dependent claim 32 should be allowable for at least the reasons discussed above with regard to claim 31.

Claims 34-37 are Allowable over Gordon

Claim 34 recites (emphasis added):

> *creating a first software development tool by integrating software development components into a software development architecture that is operable to support a plurality of different programming languages; and*
> *creating a second software development tool based on the first software development tool, wherein the second software development tool dynamically links to a binary version of the software development architecture.*

For at least the reasons discussed above with regard to claim 1, Gordon does not teach or suggest the above-cited language of claim 34. Therefore, claim 34 should be in condition for allowance.

Dependent claims 35-37 should be allowable for at least the reasons discussed above with regard to claim 34.

Claims 39-41 are Allowable over Gordon

Claim 39 recites (emphasis added):

> *dynamically linking a software development component not present in the software development architecture to a binary version of the software development architecture that is operable for the plurality of different programming languages; and*
> *creating a modified software development tool from the dynamically linked binary version and the software development component.*

For at least the reasons discussed above with regard to claim 1, Gordon does not teach or suggest the above-cited language of claim 39. Therefore, claim 39 should be in condition for allowance.

Dependent claims 40 and 41 should be allowable for at least the reasons discussed above with regard to claim 39.

Claim 43 is Allowable over Gordon

Claim 43 recites (emphasis added):

> *linking a software development component to the at least one computer-executable file using least one class extension declarations; and*
> *creating the software development tool via the linked software development component and computer-executable file.*

For at least the reasons discussed above with regard to claim 1, Gordon does not teach or suggest the above-cited language of claim 43. Therefore, claim 43 should be in condition for allowance.

Claim Rejections under 35 U.S.C. § 103 – Burger is not Prior Art

The Action rejects claims 2-3, 13-14, and 28-29 under 35 USC 103(a) as being unpatentable over Gordon in view of Burger.

As discussed above, Burger cannot be used in a rejection under § 103(a). Therefore, claims 2-3, 13-14, and 28-29 should be in condition for allowance.

Interview Request

If the claims are not found by the Examiner to be allowable, the Examiner is requested to call the undersigned attorney to set up an interview to discuss this application.

Conclusion

The claims should be allowable. Such action is respectfully requested.

Respectfully submitted,

KLARQUIST SPARKMAN, LLP

One World Trade Center, Suite 1600
121 S.W. Salmon Street
Portland, Oregon 97204
Telephone: (503) 595-5300
Facsimile: (503) 595-5301

By /Cory A. Jones/
Cory A. Jones
Registration No. 55,307